This listing of claims will replace all prior versions, and listings, of claims in the application.

Listing of Claims:

1. (Currently Amended) A method of enabling a first system to use a second system comprising:

receiving at least one data declaration pertaining to the first system;

generating interoperability information for the first and second systems comprising:

an interface definition language ("IDL") description of a portion of code on the first system for interacting with the second system;

at least one class definition pertaining to the at least one data declaration of the first system;

storing the interoperability information;

~~generating~~ receiving at the first system, a first request directed to the second system, said first request being in a form adapted for the first system but not for the second system;

performing a first conversion of said first request to produce a second request using the interoperability information, said second request being in a form adapted for said second system but not for said first system, said first conversion performed using a host initiated processing (HIP) system that includes one or more listeners for receiving the first request and one or more HIP proxies ~~that handle~~ for instantiating at least one flow control each of the at least one flow control associated with a respective requesting port for the first conversion, the number of listeners and HIP proxies to use in the first conversion is determined by an HIP runtime service based on pre-stored configuration information, the HIP system further including a first object to read conversion information from a file that explains how to translate data types from the first system to the second system to a particular flow control and a second object to access end-to-end mapping information from the file to a particular flow control;

invoking the processing of said second request by the second system;

receiving a first reply from the second system, said reply comprising an FMH7 field, said FMH7 field containing error information;

performing a second conversion of said first reply using the interoperability information to produce a second reply that comprises header information that is usable with

an Internet protocol and that represents said error information contained in said FMH7 field into header information usable with an Internet protocol; and

providing said second reply to said first system.

2. (Original) The method of claim 1, wherein said first request comprises a datum in a first format, and wherein said act of performing said first conversion comprises:

converting said datum to a second format different from said first format, said second request comprising said datum in said second format.

3. (Original) The method of claim 1, wherein said first request comprises a datum having a first type, and wherein said act of performing said first conversion comprises:

converting said datum to a second type different from said first type, said second request comprising said datum in said second type.

4. (Original) The method of claim 3, wherein said first type is supported in said first system but not in said second system.

5. (Original) The method of claim 3, wherein said second type differs structurally from said first type in at least one aspect.

6. (Original) The method of claim 1, wherein said first request comprises a call using a first mechanism to a software object in the second system, and wherein said act of performing said first conversion comprises:

converting said call for use with a second mechanism different from said first mechanism.

7. (Previously Presented) The method of claim 6, wherein said first mechanism comprises a commarea (communication area) that is used to pass a call parameter to said object and to receive a result from said object, and wherein said second mechanism comprises:

a first area that is used to pass said call parameter, or a converted call parameter that

corresponds to said call parameter, to said object;

and a second area that is used to receive said result, or a converted result that corresponds to said result, from said object.

8. (Original) The method of claim 1, wherein said first request comprises a remote call according to a first protocol, wherein said second request comprises a remote call according to a second protocol different from said first protocol, and wherein said act of performing said first conversion comprises:

preparing said second request to correspond substantively with said first request and to work in accordance with said second protocol.

9. (Original) The method of claim 8, wherein said first protocol calls for invocation to be performed with a bidirectional interaction between a caller and a callee, wherein said second protocol calls for an invocation to be performed in a unidirectional call message from said caller to said callee, and wherein said act of preparing said second request comprises:

engaging in an interaction with the caller on the first system to obtain information relating to a call;

collecting said information; and

preparing said second request using the collected information.

10. (Previously Presented) The method of claim 1, wherein said first system is adapted to communicate a remote call according to a first network protocol, and wherein said second request, and said first reply, are transmitted using a second network protocol different from said first network protocol, and wherein said acts of performing first and second conversions comprise:

including in said second request and said second reply header information that corresponds to information that is contained in requests or replies according to said first protocol.

11. (Original) The method of claim 1, wherein said first request comprises a call to a software object in said second system, and wherein the form of said first request is adapted

for making requests from the first system to a remote system that is of the same type of environment as the first system.

12-13. (cancelled)

14. (Currently Amended) A method of enabling a first software object in a first system to call a second software object in a second system, the method comprising:

evaluating first information that the first software object exposes when making a call to a remote system;

evaluating second information that the second software object exposes when receiving a call from a remote system;

receiving at least one data declaration pertaining to the first system;

generating interoperability information for the first and second systems comprising:

an interface definition language ("IDL") description of a portion of code on the first system for interacting with the second system;

at least one class definition pertaining to the at least one data declaration of the first system;

storing the interoperability information;

generating conversion information using the interoperability information and the first and second information descriptive of a process to be followed in order to convert the first information into a form compatible with the second information, said conversion information describing the conversion of an FMH7 field into header information usable with an Internet protocol;

providing the conversion information to a conversion service that uses the conversion information to convert a first call from the first object into a call in a form usable by the second object~~,~~, said conversion service using a host initiated processing (HIP) system that includes one or more listeners for receiving the first request and one or more HIP proxies that handle flow control for the first conversion, the number of listeners and HIP proxies to use in the first conversion is determined by an HIP runtime service based on pre-stored configuration information, the HIP system further including a first object to read conversion information from a file that explains how to translate data types from the first system to the

second system to a particular flow control and a second object to access end-to-end mapping information from the file to a particular flow control.

15. (Original) The method of claim 14, wherein the first information comprises a call parameter in a first format, wherein the second information comprises a call parameter in a second format, and wherein the act of generating conversion information comprises:

generating code or data that describes how to convert a call parameter from the first format to the second format.

16. (Original) The method of claim 14, wherein the first information comprises a call parameter of a first data type which is not usable by the second software object, and wherein the act of generating conversion information comprises:

generating a second data type that corresponds to the first data type and which is usable by the second software object;

generating code or data that describes how to convert data of the first data type to the second data type.

17. (Original) The method of claim 14, wherein the first information comprises a return value in a first form, wherein the second information comprises a return value in a second form different from said first form, and wherein the act of generating conversion information comprises:

generating code or data that describes how to convert data in said first form to said second form.

18. (Original) The method of claim 14, wherein said first software object makes a call to a remote system according to a first programming model, wherein said second data object receives a call from a remote system according to a second programming model, and wherein the act of generating conversion information comprises:

generating code or data that indicates which programming model the first software object uses to make a remote call.

19. (Original) The method of claim 18, wherein the act of generating conversion information comprises:

generating code or data that describes at least one customization in converting from the first programming model to the second programming model.

20. (Original) The method of claim 14, wherein the act of generating conversion information comprises:

generating a transaction initiation message that is used in invoking the second software object or in reply to the first software object.

21-22 (cancelled).

23. (Currently Amended) A system to enable a first software object in a first environment to call a second software object in a second environment, the system comprising:

hardware comprising at least one processor;

a component builder module that operates on said hardware, said component builder:

receiving at least one data declaration pertaining to the first system;

generating interoperability information for the first and second systems comprising:

an interface definition language ("IDL") description of a portion of code on the first system for interacting with the second system;

at least one class definition pertaining to the at least one data declaration of the first system;

storing the interoperability information;

a service object that executes on said hardware and that receives a first request from the first software object, converts the first request into a second request which is in a form usable by the second software object, and presents the second request to the second software object using the interoperability information;

a host initiated processing (HIP) system that includes one or more listeners for receiving the first request and one or more HIP proxies that handle flow control, the number

of listeners and HIP proxies to used is determined by an HIP runtime service based on pre-stored configuration information, the HIP system further including a first object to read conversion information from a .TIM file that explains how to translate data types from the first system to the second system to a particular flow control and a second object to access end-to-end mapping information from the .TIM file to a particular flow control, the HIP system further including first and second conversion components, the first conversion component handling conversion issues related to aggregate data types and the second conversion component handling conversions issues related to primitive data types; and

an error handling object that executes on said hardware and that receives an indication of an error from the second software object and packages the error into a form usable by the first environment or the first software object, wherein the indication of said error comprises an FMH7 field, and wherein said error handling object creates header information representative of the contents of said FMH7 field, said header information being adapted for use with an Internet protocol.

24. (Original) The system of claim 23, wherein the service object further receives a first reply from the second software object, converts the first reply to a second reply which is in a form usable by the first software object, and provides the second reply to the first software object.

25. (Previously Presented) The system of claim 23, wherein the service object comprises:

a listener object that executes on said hardware and that detects that a contact regarding the first request has been made by the first software object.

26. (Previously Presented) The system of claim 23, wherein the service object comprises:

a queuing object that executes on said hardware and that queues at least one of connections and requests from the first system.

27. (Previously Presented) The system of claim 23, wherein the service object comprises:

a transit object that executes on said hardware and that receives information related to the first request from the first software object and prepares the information into a form that can be used for a call to the second software object.

28. (Previously Presented) The system of claim 27, wherein the service object further comprises:

an invocation object that executes on said hardware and that lays out the information prepared by the transit object into a form that can be used for a call to the second software object, and that uses the laid out information to invoke the second software object.

29. (Previously Presented) The system of claim 23, wherein the service object comprises:

a flow control object that executes on said hardware and that manages the interaction of one or more components involved in the conversion of the first request into the second request.

30-31 (cancelled).

32. (Currently Amended) A computer-readable storage medium encoded with computer-executable instructions to facilitate interoperability between a first system and a second system, the instructions being adapted to perform acts comprising:

receiving at least one data declaration pertaining to the first system;

generating interoperability information for the first and second systems comprising:

an interface definition language ("IDL") description of a portion of code on the first system for interacting with the second system;

at least one class definition pertaining to the at least one data declaration of the first system;

storing the interoperability information;

~~generating~~ receiving a first call from a first software object in the first system to a second software object in the second system, the first call being in a format that is not compatible with the second system;

converting the first call into a second call using the interoperability information, the second call being in a format that is compatible with the second, said converting performed using a host initiated processing (HIP) system that includes one or more listeners for receiving the first request and one or more HIP proxies that handle flow control for the first conversion, the number of listeners and HIP proxies to use in the first conversion is determined by an HIP runtime service based on pre-stored configuration information, the HIP system further including a first object to read conversion information from a file that explains how to translate data types from the first system to the second system to a particular flow control and a second object to access end-to-end mapping information from the file to a particular flow control, the HIP system further including first and second conversion components, the first conversion component handling conversion issues related to aggregate data types and the second conversion component handling conversions issues related to primitive data types;

converting an FMH7 field generated by said first system into header information usable by an Internet protocol;

invoking the second software object using the second call;

listening for a connection from the first system; and

receiving information related to the first call in response to the connection, wherein the first call comprises data in a first form; and

converting the data from the first form into a second form usable by the second software object.

33. (cancelled)

34. (cancelled)

35. (Previously Presented) The computer-readable storage medium of claim 32, wherein the second software object provides a first reply in response to being called, and

wherein the instructions are adapted to perform acts further comprising:

converting the first reply into a second reply, the second reply being in a form that is compatible with the first software object or the first system, the first reply being in a form that is not compatible with the first software object or the first system.

36 (Previously Presented) The computer-readable storage medium of claim 32, wherein the second software object generates error information in response to being called, and wherein the instructions are adapted to perform acts further comprising:

converting said error information into a format compatible with the first software object or the first system, or into a format compatible with a communication protocol employed by the first system.

37. (Canceled)